TRAN-045/ACM/JSG

United States Patent Application

For

SUPPORTING SPECULATIVE MODIFICATION IN A DATA CACHE

Inventors:

Bill Rozas
Alexander Klaiber
David Dunn
Paul Serris
Lacky Shah

Prepared by:

WAGNER, MURABITO & HAO LLP
Two North Market Street
Third Floor
San Jose, California 95113

(408) 938-9060

CONFIDENTIAL

SUPPORTING SPECULATIVE MODIFICATION IN A DATA CACHE

BACKGROUND OF THE INVENTION

FIELD OF THE INVENTION

The present invention generally relates to data caches. More particularly, the present invention relates to the field of supporting speculative modification in a data cache.

RELATED ART

A data cache interacts with a processor to increase system performance. However, if the processor is speculatively executing instructions, a traditional data cache is unable to properly deal with speculative modifications.

SUMMARY OF THE INVENTION

Method and system for supporting speculative modification in a data cache are provided and described.

BRIEF DESCRIPTION OF THE DRAWINGS

The accompanying drawings, which are incorporated in and form a part of this specification, illustrate embodiments of the invention and, together with the description, serve to explain the principles of the present invention.

Figure 1 illustrates a system in accordance with a first embodiment of the present invention.

Figure 2 illustrates a state diagram of a data cache in accordance with a first embodiment of the present invention.

Figure 3 illustrates a system in accordance with a second embodiment of the present invention.

Figure 4 illustrates a first state diagram of a speculative cache buffer in accordance with a second embodiment of the present invention.

Figure 5 illustrates a second state diagram of a speculative cache buffer in accordance with a second embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

Reference will now be made in detail to embodiments of the present invention, examples of which are illustrated in the accompanying drawings. While the invention will be described in conjunction with these embodiments, it will be understood that they are not intended to limit the invention to these embodiments. On the contrary, the invention is intended to cover alternatives, modifications and equivalents, which may be included within the spirit and scope of the invention as defined by the appended claims. Furthermore, in the following detailed description of the present invention, numerous specific details are set forth in order to provide a thorough understanding of the present invention. However, it will be recognized by one of ordinary skill in the art that the present invention may be practiced without these specific details.

Figure 1 illustrates a system 100 in accordance with a first embodiment of the present invention. As illustrated in Figure 1, the system 100 includes a processor 10 and a data cache 20.

The processor 10 is able to speculatively execute instructions. If the processor 10 speculatively executes instructions to a particular instruction boundary without generating errors, the speculative store operations to the data cache 20 can be made permanent with a commit operation. However, if errors occur before reaching the particular instruction boundary, the speculative store operations to the data cache 20 have to be undone with a rollback operation.

The data cache 20 includes a plurality of cache lines 25. Each cache line includes a state indicator 27 for indicating anyone of a plurality of states. The plurality of states include an invalid state, a valid state, a dirty state, and a speculative state. The invalid state indicates that the respective cache line is not being used. The valid state indicates that the respective cache line has clean data. The dirty state indicates that the respective cache line has dirty data (or the most recent data compared to other memory components such as L2 data cache, main memory, etc.). The speculative state enables keeping track of speculative modification to data in said respective cache line. The speculative state enables a speculative modification to the data in the respective cache line to be made permanent in response to a commit operation. Moreover, the speculative state enables the speculative modification to the data in the respective cache line to be undone in response to a rollback operation. Cache lines having the speculative state cannot be drained to other memory components such as L2 data cache, main memory, etc.

Figure 2 illustrates a state diagram of a data cache in accordance with a first embodiment of the present invention. As described above, a cache line can have an invalid state I, a valid state V, a dirty state D, or a speculative state S. (For clarity, state transitions from V, D, and S states to the I state, corresponding to the traditional operation of the data cache 20 evicting a cache line 25, have been omitted in Figure 2)

Invalid state I

Assuming the cache line is in the invalid state I, there are several possibilities for this cache line. If a non-speculative store is performed by the processor 10 (Figure 1), the cache line moves to the dirty state D. If data is loaded from memory components such as L2 data cache, main memory, etc., the cache line moves to the valid state V, where the data is clean (has same version as the memory components such as L2 data cache, main memory, etc.). If a speculative store is performed by the processor 10 (Figure 1), the cache line moves to the speculative state S.

Valid state V

Assuming the cache line is in the valid state V, there are several possibilities for this cache line. If a non-speculative store is performed by the processor 10 (Figure 1), the cache line moves to the dirty state D. If a speculative store is performed by the processor 10 (Figure 1), the cache line moves to the speculative state S.

Dirty state D

Assuming the cache line is in the dirty state D, there are several possibilities for this cache line. If a speculative store is performed by the processor 10 (Figure 1) to this cache line, the cache line is first written back to a memory component such as L2 data cache, main memory, etc., thus preserving the cache line data as of before the speculative modification. Then, the speculative store is performed, moving the cache line to the speculative state S.

Speculative State S

Assuming the cache line is in the speculative state S, there are several possibilities for this cache line. If a commit operation is performed, the cache line moves to the dirty state D. If a rollback operation is performed, the cache line moves to the invalid state I.

Figure 3 illustrates a system 300 in accordance with a second embodiment of the present invention. The system 300 includes a processor 10, a data cache 20, and a speculative cache buffer 50. The discussion with respect to the processor 10 and the data cache 20 is equally applicable to Figure 3.

The speculative cache buffer 50 receives cache lines which have the speculative state S or the dirty state D and are evicted or drained from the data cache 20. Hence, the data cache 20 can send cache lines having the speculative state S or the dirty state D to the speculative cache buffer 50 and retrieve them when necessary.

Moreover, the speculative cache buffer 50 has a plurality of cache lines 55. Each cache line 55 includes a state indicator 57 for indicating anyone of a plurality of states. The plurality of states includes an invalid state, a dirty state, and a speculative state. In one embodiment, the speculative cache buffer 50 is fully associative.

The data cache 20 can drain cache lines that are in the dirty state D or the speculative state S to the speculative cache buffer 50. Moreover, the speculative cache buffer 50 can drain cache lines that are in the dirty state D to a memory component such as L2 data cache, main memory, etc.

Figure 4 illustrates a first state diagram of a speculative cache buffer in accordance with a second embodiment of the present invention. As described above, a cache line can have an invalid state I, a dirty state D, or a speculative state S.

Invalid state I

Assuming the cache line is in the invalid state I, there are several possibilities for this cache line. If the data cache 20 evicts a cache line having the dirty state D, the cache line moves to the dirty state D. If the data cache 20 evicts a cache line having the speculative state S, the cache line moves to the speculative state S.

Dirty state D

Assuming the cache line is in the dirty state D, there are several possibilities for this cache line. If the speculative cache buffer 50 drains the cache line having the dirty state D to a memory component such as L2 data cache, main memory, etc., the cache line moves to the invalid state I. In case the data cache 20 requests the cache line back, the cache line moves to the invalid state I in the speculative cache buffer 50.

Speculative State S

Assuming the cache line is in the speculative state S, there are several possibilities for this cache line. If a commit operation is performed, the cache line moves to the dirty state D. If a rollback operation is performed, the cache line moves to the invalid state I. In case the data cache 20 requests the cache line back, the cache line moves to the invalid state I in the speculative cache buffer 50.

It is possible that multiple versions of a cache line in the dirty state may exist in the speculative cache buffer 50. For instance, the data cache 20 may drain the cache line having the dirty state to the speculative cache buffer 50 because a speculative store has to be performed to the cache line in the data cache 20. If the cache line having the speculative state is later drained to the speculative cache buffer 50 and if a commit operation is performed, then the speculative cache buffer 50 would have two cache lines with different versions of the data, whereas only one version of the data needs to be drained to a memory component such as L2 data cache, main memory, etc.

In an alternate embodiment of the speculative cache buffer 50, the plurality of states also includes a commit-kill state, in addition to the invalid state, the dirty state, and the speculative state. The commit-kill state indicates that the data cache 20 has evicted the respective cache line having the dirty state in response to a speculative modification operation (or speculative store) to the respective cache line in the data cache 20. The commit-kill state reduces the number of copies of a cache line in the dirty state and saves bandwidth in case of the commit operation, as detailed below.

Figure 5 illustrates a second state diagram of a speculative cache buffer in accordance with a second embodiment of the present invention. As described above, a cache line can have an invalid state I, a dirty state D, a commit-kill state K, or a speculative state S.

Invalid state I

Assuming the cache line is in the invalid state I, there are several possibilities for this cache line. If the data cache 20 evicts a cache line having the dirty state D but not due to a speculative store operation, the cache line moves to the dirty state D. If the data cache 20 evicts a cache line having the speculative state S, the cache line moves to the speculative state S. If the data cache 20 evicts a cache line having the dirty state D in response to a speculative store operation to that cache line, the cache line moves to the commit-kill state K.

Dirty state D

Assuming the cache line is in the dirty state D, there are several possibilities for this cache line. If the speculative cache buffer 50 drains the cache line having the dirty state D to a memory component such as L2 data cache, main memory, etc., the cache line moves to the invalid state I. In case the data cache 20 requests the cache line back, the cache line moves to the invalid state I in the speculative cache buffer 50.

Speculative State S

Assuming the cache line is in the speculative state S, there are several possibilities for this cache line. If a commit operation is performed, the cache line moves to the dirty state D. If a rollback operation is performed, the cache line moves to the invalid state I. In case the data cache 20 requests the cache line back, the cache line moves to the invalid state I in the speculative cache buffer 50.

Commit-kill State K

Assuming the cache line is in the commit-kill state K, there are several possibilities for this cache line. If a commit operation is performed, the cache line moves to the invalid state I. If a rollback operation is performed, the cache line moves to the dirty state D. If the speculative cache buffer 50 drains the cache line having the commit-kill state K to a memory component such as L2 data cache, main memory, etc., the cache line moves to the invalid state I.

The foregoing descriptions of specific embodiments of the present invention have been presented for purposes of illustration and description. They are not intended to be exhaustive or to limit the invention to the precise forms disclosed, and many modifications and variations are possible in light of the above teaching. The embodiments were chosen and described in order to best explain the principles of the invention and its practical application, to thereby enable others skilled in the art to best utilize the invention and various embodiments with various modifications as are suited to the particular use contemplated. It is intended that the scope of the invention be defined by the Claims appended hereto and their equivalents.